Code Change Helper

Note: the examples are for CMR. User your project name where appropriate.

Code Maintainer

Before Starting Work

1. Create a local branch for the code. Name it using the jira ticket number: CMR-1234
    - git checkout -b CMR-1234
2. Push the local branch to git (you may elect to do this at a later point)
    - git push -u origin HEAD
3. Make your code change and test it
4. Push your changes on your local branch to the remote branch. (Assumes you already did the push to create the remote branch on step 2)
    - git push

Pulling from master

As you work on your feature branch another developer may merge something to master. You should keep up to date with that code so...

1. Commit code
2. git fetch origin master
3. git merge FETCH_HEAD
4. If you have conflicts
    a. Correct code
    b. Run your tests to success
    c. Commit code
5. git push -u origin HEAD

After Implementing a Feature

- Commit and push your code changes in the branch.
- Logon to the Git repo in Bitbucket (https://git.earthdata.nasa.gov)
    - In the Common Metadata Repository Bitbucket Project then select the repo
- Create a pull request
    - Click the Pull Request button in upper right
    - Select source branch (CMR-1234)
    - Destination branch will be master
    - Update title and description if necessary
    - Choose 1 or more reviewers

Resolving merge conflicts or repairing mistakes AFTER the Pull Request is submitted and before it is approved and merged

The pull request will indicate if there are merge conflicts and how to resolve them before merging. A similar approach applies if there are mistakes detected after the pull request has been submitted or the code maintainer forgot to run the tests before submitting the pull request, runs them after the pull request submission and there are failures that need additional local changes in the source branch.

- Fetch the changes that you must reconcile with (only do this if there are merge conflicts):
  git fetch origin master
- Checkout your source branch and merge changes from the target branch (only do this if there are merge conflicts). Resolve merge conflicts:
  git checkout CMROS-1234
  git merge FETCH_HEAD
- After merge conflicts are resolved OR mistakes are rectified via local changes in the source branch, stage the changes, commit the changes and push. This is the only step that must be executed if local changes are needed after the pull request submission:
  git commit
  git push origin HEAD

The pull request is now updated with the resolved merge conflicts or the local changes needed to correct mistakes detected after the pull request was submitted.

After a Pull Request has been declined

The reviewer will decline a pull request if there are comments that require changes.

- Make fixes based on comments.
- Commit and push them to the branch.
- Go to Bitbucket and re-open the pull request.
    - The reviewer should see this and be able to review the changes.

After a Pull Request has been approved

After the pull request has been approved by the acting development lead and at least one other developer the code can be merged. The code maintainer is responsible for this.

Reviewer (Training - Code Reviews with Bitbucket)

- Got to git repo in Bitbucket and view the pull request
- Review code and make comments in diff.
- If code is good approve it. The code maintainer will merge it.
- If code needs changes decline it.
    - The code maintainer will need to update the code and then issue a new pull request.